Analysis Report

OBJECTIVE:

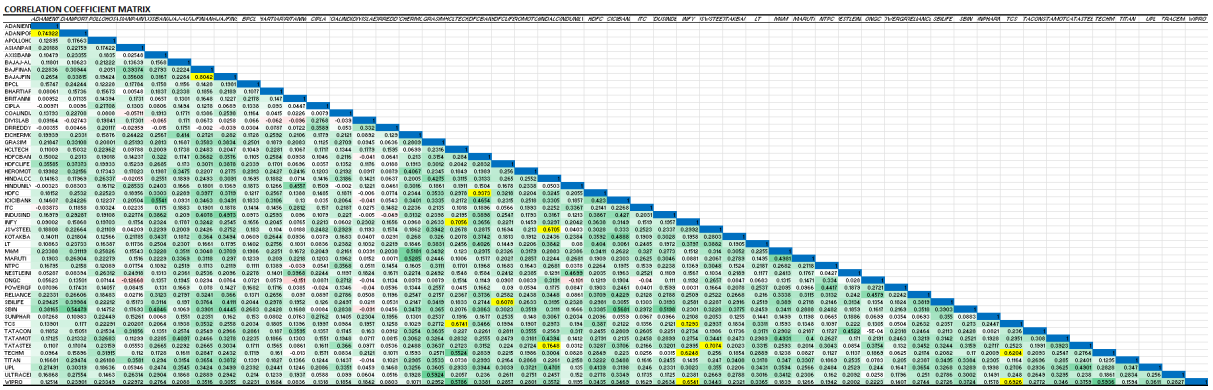
The objective of this assignment is to analyse and implement a pair trading strategy, choosing two stocks from the NIFTY 50 index basis their correlation coefficient over the past year. Then collecting historical data, implementing the strategy and evaluating its performance using different metrics and carrying out a sensitivity analysis on the basis of different parameters used in the strategy and optimising the strategy for best combination.

APPROACH:

1. **Correlation Analysis**:
   For this, MS-Excel workbook (stock_data.xlsx) was used to calculate the various correlation coefficients and create a matrix for the same.
   First, downloading historical data for all NIFTY 50 stocks over the past year, the 'daily closing prices' was stored in a sheet (stock_data_Closing.csv).
   The daily returns were calculated and stored in another sheet (stock_data_Daily.csv) and using the inbuilt function to calculate the correlation, a correlation matrix was created in another sheet (stock_data_Correlation.csv).
   Using the matrix, stock pairs with correlation coefficient over 0.6 were selected, out of which the pair which were distinct with a considerable coefficient was chosen for the assignment.
   The stocks chosen were Hindalco and Tata Steel.
2. **Pair Trading Strategy Implementation**:
   For this, we use a jupyter notebook and write the python code.
   First, downloading data for the chosen stocks and calculating the spread, differentials, price ratio, correlation coefficient and z-score. Then these were plotted.
   The strategy is then developed and implemented by defining parameters such as lookback period, entry threshold and exit threshold, and we find the positions calculated by the strategy. This is then plotted.
   The strategy is evaluated by finding the portfolio value and cumulative value at various stages of the strategy. Another parameter trading cost is also introduced, and the cumulative returns are plotted. Also, the sharpe ratio and maximum drawdown are calculated as suitable metrics.
3. **Sensitivity Analysis**:
   For this, taking a range of values for the various parameters used above, a simple model is formulated to calculate the different metrics as calculated above. This outputs the various combinations of parameters and their corresponding metrics. The criteria for optimal output is also enforced to return the optimal combination of parameters and their corresponding metrics.

CORRELATION ANALYSIS:

The correlation coefficient matrix,



From the correlation coefficient matrix obtained, these were the pair of stocks with highest coefficient values.

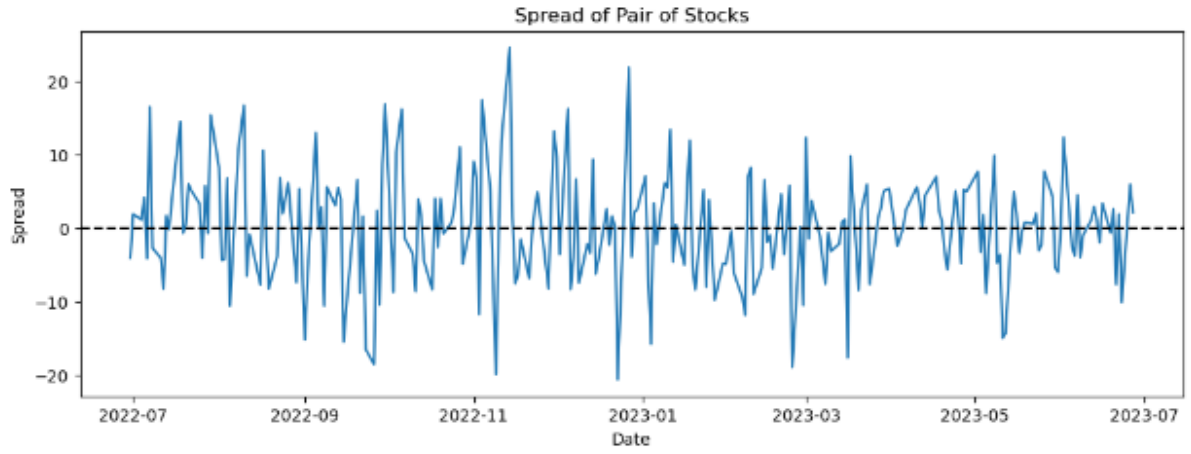
Stock1	Stock2	Correlation	Sector	Comment
HDFCBANK	HDFC	0.93732	Banking/Finance	Subsidiary and Parent, Merger
BAJFINANCE	BAJAJFINSV	0.804188	Finance	Subsidiary and Parent
HINDALCO	TATASTEEL	0.764835	Metals	Same Sector, Leaders
ADANIENT	ADANIPORTS	0.749223	Services	Sisters
INFY	TCS	0.729323	Technology	Same Sector, Leaders
JSWSTEEL	TATASTEEL	0.707401	Metals	Same Sector
HCLTECH	INFY	0.70556	Technology	Same Sector
TCS	WIPRO	0.69262	Technology	Same Sector, Leaders
HCLTECH	TCS	0.674142	Technology	Same Sector
HINDALCO	JSWSTEEL	0.670482	Metals	Same Sector
INFY	WIPRO	0.654062	Technology	Same Sector, Leaders
INFY	TECHM	0.624817	Technology	Same Sector
TCS	TECHM	0.620431	Technology	Same Sector
HDFCLIFE	SBILIFE	0.607826	Insurance	Same Sector

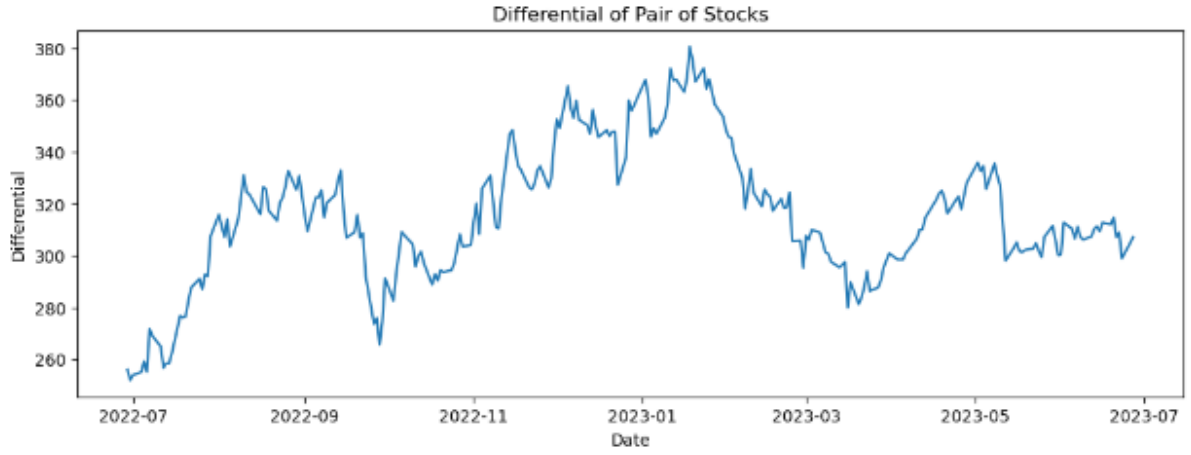
From this data, the stocks chosen were Hindalco and Tata Steel (highlighted). This came about considering that the other pairs having correlation greater or equal to it were parent and subsidiary companies or sister companies. Only this and the Infosys and TCS pair interested me as they were wholly different companies with different management and leaderships, belonging to the same sectors and being their respective industry's leaders for a considerable time now. In spite of being in fierce competition with each other, they had high correlation values. Finally, the pair with the higher correlation coefficient, ie, Hindalco and Tata Steel was chosen.

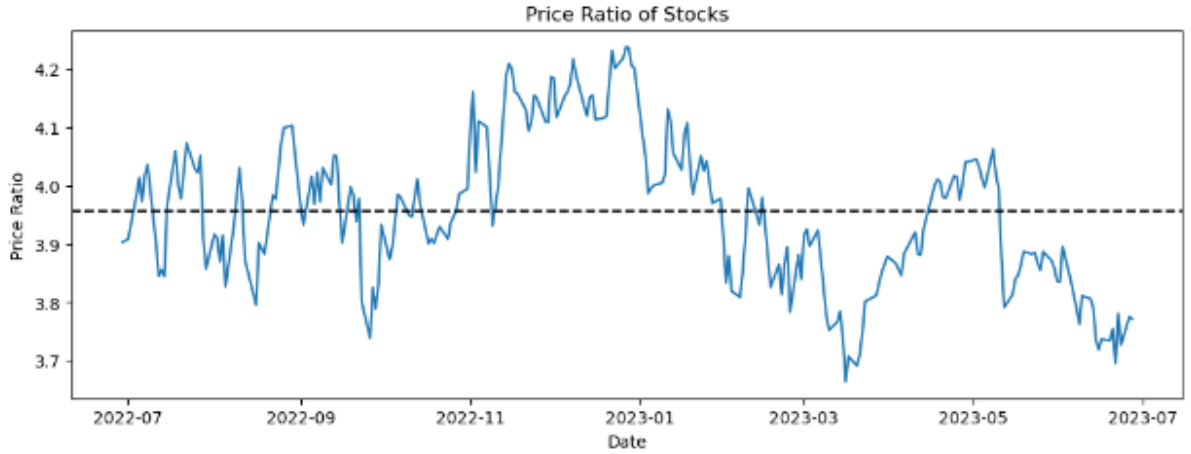
PAIR TRADING STRATEGY IMPLEMENTATION:

Calculating the required terms and plotting the relevant graphs, the strategy has been implemented. Here are the important code snippets,

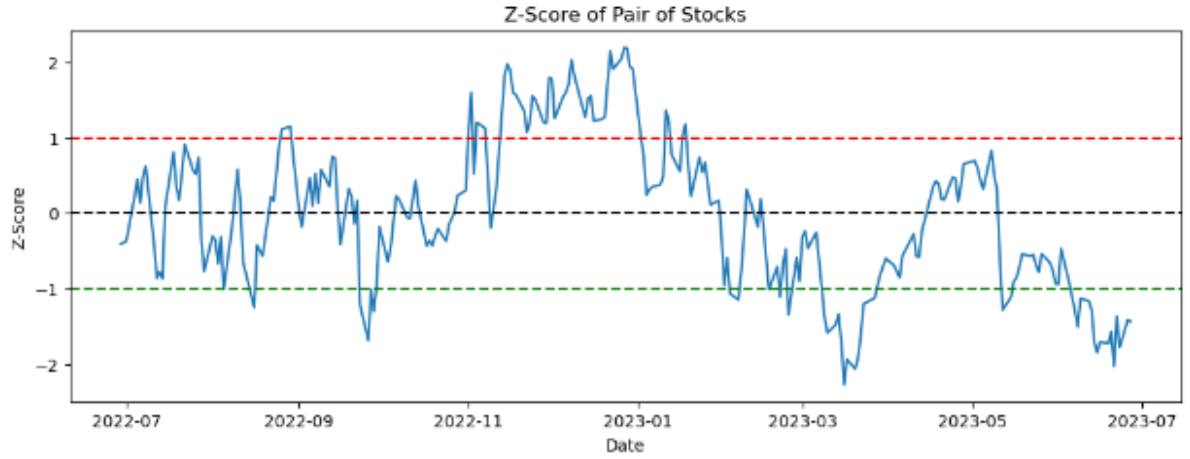
```
# Spread is difference between closing values of the stocks
data['Spread'] = data['H_Closing'] - data['T_Closing']

# Differential is difference between closing prices of the stocks
data['Differential'] = data['HINDALCO'] - data['TATASTEEL']

# Price Ratio is the ratio of the closing prices of the stocks
data['Price_Ratio'] = data['HINDALCO'] / data['TATASTEEL']
```







```
# Calculate the z-score
mean = np.mean(data['Price_Ratio'])
std = np.std(data['Price_Ratio'])
data['z-score'] = (data['Price_Ratio'] - mean) / std
```

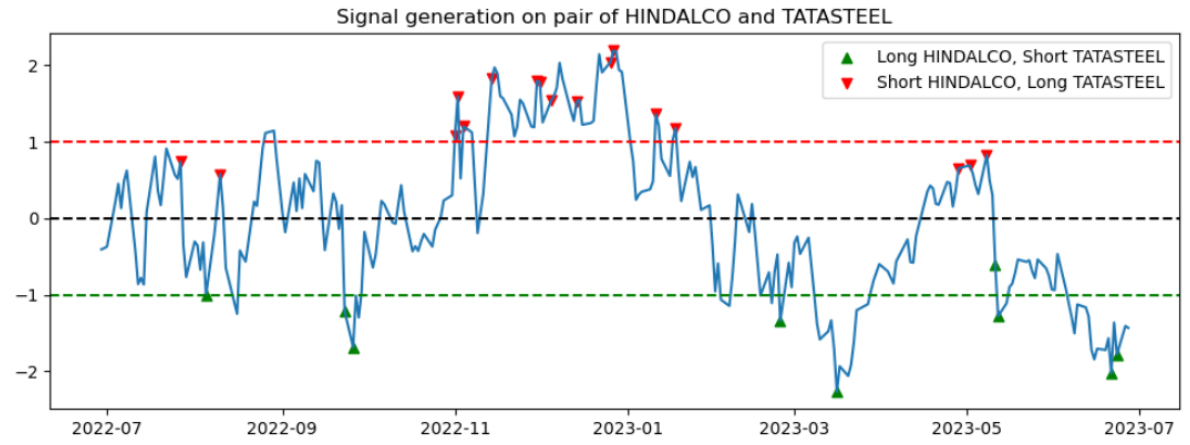


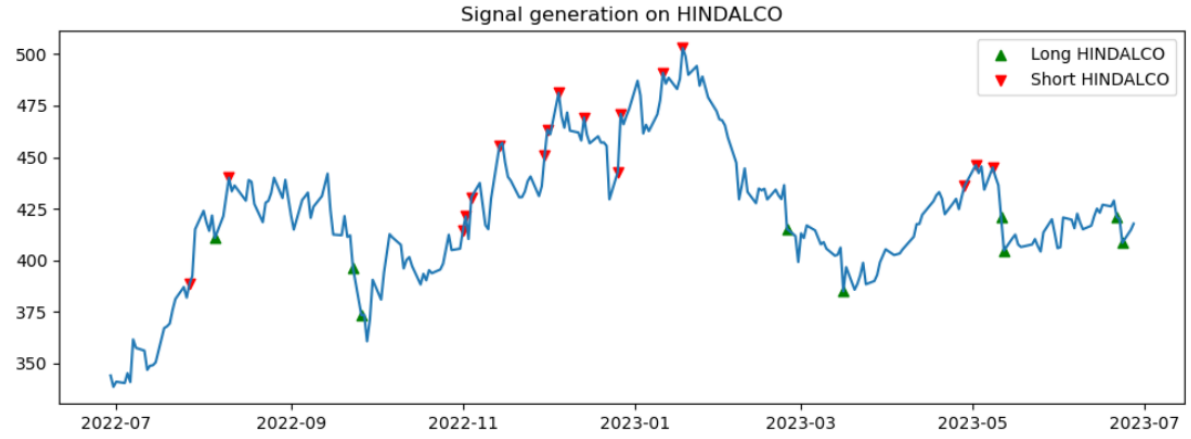

```
# Correlation coefficient using daily returns
Correlation = np.corrcoef(data['H_DailyRet'].tail(246), data['T_DailyRet'].tail(246))[1, 0]
print('Correlation between HINDALCO and TATASTEEL using daily returns:', Correlation)
```

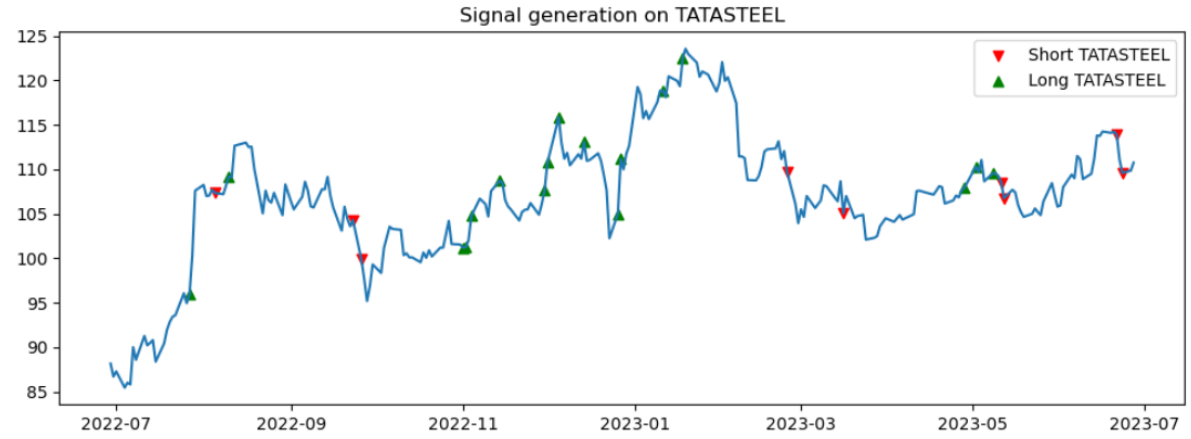
```
Correlation between HINDALCO and TATASTEEL using daily returns: 0.7647884797470316
```

```
# Developig the entry/exit rules and buy/sell signals
lookback_period = 14
entry_threshold = 1.5
exit_threshold = 0.7

Spread_Mean = data['Spread'].rolling(window=lookback_period).mean()
Spread_Std = data['Spread'].rolling(window=lookback_period).std()

# Iterate over the spread data
data['Position'] = 0  # 1: Long HINDALCO, short TATASTEEL = Buy | -1: short HINDALCO, long TATASTEEL = Sell | 0: no position
data['Action'] = 'No action'
Entry_Price = 0

# Check if the spread crosses the entry threshold in the desired direction
data.loc[(data['Spread'] < (Spread_Mean - (entry_threshold * Spread_Std))) & (data['z-score'] < -0.5) & (data['Position'] == 0),
data.loc[(data['Spread'] > (Spread_Mean + (exit_threshold * Spread_Std))) & (data['z-score'] > 0.5) & (data['Position'] == 0), '
Entry_Price = data.query('Position != 0')['Differential'][0]

# Perform trading actions based on the position
data.loc[data['Position'] == 1, 'Action'] = 'Long Hindalco, Short TataSteel'  # Take action for long position: buy HINDALCO, sho
data.loc[data['Position'] == -1, 'Action'] = 'Short Hindalco, Long TataSteel'  # Take action for short position: short HINDALCO,

print('Entry Price =', Entry_Price)
```

```
Entry Price = 292.7099914550781
```








```python
# Calculating portfolio value and cumulative returns
capital = 100000
trading_cost = 0.001

n1 = (capital // 2) // data['HINDALCO'][0]
n2 = (capital // 2) // data['TATASTEEL'][0]
capital -= n1 * data['HINDALCO'][0]
capital -= n2 * data['TATASTEEL'][0]

if np.less(data[data['Position'] == -1]['Position'].count(), data[data['Position'] == 1]['Position'].count()):
    Total_Trades = data[data['Position'] == -1]['Position'].count()
else: Total_Trades = data[data['Position'] == 1]['Position'].count()

Current_Value = []
Returns = []

#Iterate over positions
for i in range(0, len(data['Position'])):
    if data['Position'][i] == -1:  # short HINDALCO, long TATASTEEL
        capital += n1 * data['HINDALCO'][i-1]
        n1 = 0
        n2 += capital // data['TATASTEEL'][i-1]
        capital -= (capital // data['TATASTEEL'][i-1]) * data['TATASTEEL'][i-1]
    elif data['Position'][i] == 1:  # Long HINDALCO, short TATASTEEL
        capital += n2 * data['TATASTEEL'][i-1]
        n2 = 0
        n1 += capital // data['HINDALCO'][i-1]
        capital -= (capital // data['HINDALCO'][i-1]) * data['HINDALCO'][i-1]

    Current_Value.append((n1 * data['HINDALCO'][i-1] + n2 * data['TATASTEEL'][i-1]) + capital)
    Returns.append((((Current_Value[i] - 100000) / 100000) - (Total_Trades * trading_cost)) * 100)

data['Portfolio_Value'] = Current_Value
data['Cumulative_Returns'] = Returns
Cumulative_Return = data['Cumulative_Returns'][246]

print('Portfolio Value =', data['Portfolio_Value'][246])
print('Cumulative Return =', Cumulative_Return, '%')
```

```
Portfolio Value = 123804.20993804932
Cumulative Return = 22.904209938049316 %
```


```python
# Calculating sharpe ratio
risk_free_rate = 0.05
Average_Return = data['Cumulative_Returns'].mean()
Volatality = data['Cumulative_Returns'].std()

if Volatality != 0:
    Sharpe_Ratio = (Average_Return - risk_free_rate) / Volatality
else: Sharpe_Ratio = 0

print('Sharpe Ratio =', Sharpe_Ratio)
```

```
Sharpe Ratio = 2.4513403546351795
```

```python
# Calculating maximum drwadown
Peak = np.max(data['Portfolio_Value'])
for i in range(0, len(data['Portfolio_Value'])):
    if data['Portfolio_Value'][i] == Peak:
        Trough = np.min(data['Portfolio_Value'].tail(-i))

Max_Drawdown = (Peak - Trough) / Peak * 100

print('Maximum Drawdown =', Max_Drawdown, '%')
```

```
Maximum Drawdown = 20.004571891276665 %
```

SENSITIVITY ANALYSIS:

Writing the code for sensitivity analysis applying the same code as above after designing the model structure to formulate the metrics as per the different parameter combinations.

```
# Sensitivity Analysis
thresholds = [0.5, 0.7, 1.0, 1.5, 2.5]
lookback_periods = [10, 14, 30, 60]
trading_costs = [0.001, 0.005, 0.01]
risk_free_rates = [0.02, 0.05, 0.1]

# Define target variable and metrics and choose the target variable to optimize
target_variable_1 = 'Portfolio Value'
target_variable_2 = 'Cumulative Return'
target_variable_3 = 'Sharpe Ratio'
target_variable_4 = 'Maximum Drawdown'
best_metric_1 = -float('inf')  # Initialize the best metric value
best_metric_2 = -float('inf')
best_metric_3 = -float('inf')
best_metric_4 = float('inf')
best_parameters = None  # Initialize the best parameter combination

# Iterate over parameter combinations
for entry_threshold, exit_threshold, lookback_period, trading_cost, risk_free_rate in it.product(thresholds, thresholds, lookbac
    # Calculate the spread, mean, and standard deviation
    spread_mean = data['Spread'].rolling(window=lookback_period).mean()
    spread_std = data['Spread'].rolling(window=lookback_period).std()

    position = np.arange(247)  # 1: Long HINDALCO, short TATASTEEL = Buy | -1: short HINDALCO, Long TATASTEEL = Sell | 0: no pos
    position.fill(0)

    for i in range(0, len(data['Spread'])):
        # Check if the spread crosses the entry threshold in the desired direction
        if (data['Spread'][i] < (spread_mean[i] - (entry_threshold * spread_std[i]))) and (data['z-score'][i] < 0) and (position
            position[i] = 1  # Enter Long position: Long HINDALCO, short TATASTEEL
        if (data['Spread'][i] > (spread_mean[i] + (exit_threshold * spread_std[i]))) and (data['z-score'][i] > 0) and (position[
            position[i] = -1 # Enter short position: short HINDALCO, Long TATASTEEL

    # Calculating portfolio value and cumulative return
    cap = 100000

    N1 = (cap // 2) // data['HINDALCO'][0]
    N2 = (cap // 2) // data['TATASTEEL'][0]
    cap -= N1 * data['HINDALCO'][0]
    cap -= N2 * data['TATASTEEL'][0]

    if np.less(np.count_nonzero(position == -1), np.count_nonzero(position == 1)):
        trades = np.count_nonzero(position == -1)
    else: trades = np.count_nonzero(position == 1)

    value = []
    returns = []

    for i in range(0, len(position)):
        if position[i] == -1:  # short HINDALCO, long TATASTEEL
            cap += N1 * data['HINDALCO'][i-1]
            N1 = 0
            N2 += cap // data['TATASTEEL'][i-1]
            cap -= (cap // data['TATASTEEL'][i-1]) * data['TATASTEEL'][i-1]
        elif position[i] == 1:  # Long HINDALCO, short TATASTEEL
            cap += N2 * data['TATASTEEL'][i-1]
            N2 = 0
            N1 += cap // data['HINDALCO'][i-1]
            cap -= (cap // data['HINDALCO'][i-1]) * data['HINDALCO'][i-1]

        value.append((N1 * data['HINDALCO'][i-1] + N2 * data['TATASTEEL'][i-1]) + cap)
        returns.append((((value[i] - 100000) / 100000) - (trades * trading_cost)) * 100)

        portfolio_value = value[-1]
        cumulative_return = returns[-1]
```

```
    # Calculating sharpe ratio
    avg_return = np.mean(returns)
    volatality = np.std(returns)

    if volatality != 0:
        sharpe_ratio = (avg_return - risk_free_rate) / volatality
    else: sharpe_ratio = 0

    # Calculating maximum drwadown
    df = pd.Series(value)
    peak = np.max(df)
    for i in range(0, len(df)):
        if df[i] == peak:
            trough = np.min(df.tail(-i))

    maximum_drawdown = (peak - trough) / peak * 100

    # Check if the current metric value is better than the previous best value
    if portfolio_value > best_metric_1:
        best_metric_1 = portfolio_value
        best_parameters = (entry_threshold, exit_threshold, lookback_period, trading_cost, risk_free_rate)
    if cumulative_return > best_metric_2:
        best_metric_2 = cumulative_return
        best_parameters = (entry_threshold, exit_threshold, lookback_period, trading_cost, risk_free_rate)
    if sharpe_ratio > best_metric_3:
        best_metric_3 = sharpe_ratio
        best_parameters = (entry_threshold, exit_threshold, lookback_period, trading_cost, risk_free_rate)
    if maximum_drawdown < best_metric_4:
        best_metric_4 = maximum_drawdown
        best_parameters = (entry_threshold, exit_threshold, lookback_period, trading_cost, risk_free_rate)

    # Printing parameters and their repsective metrics
    print('Parameters:')
    print('Entry_Threshold =', entry_threshold)
    print('Exit_Threshold =', exit_threshold)
    print('Lookback Period =', lookback_period)
    print('Trading Cost =', trading_cost)
    print('Risk Free Rate =', risk_free_rate)
    print('Metrics:')
    print(target_variable_1, '=', portfolio_value)
    print(target_variable_2, '=', cumulative_return)
    print(target_variable_3, '=', sharpe_ratio)
    print(target_variable_4, '=', maximum_drawdown)
    print('')
```

```
Parameters:
Entry_Threshold = 0.5
Exit_Threshold = 0.5
Lookback Period = 10
Trading Cost = 0.001
Risk Free Rate = 0.02
Metrics:
Portfolio Value = 114527.41246032715
Cumulative Return = 10.82741246032715
Sharpe Ratio = 1.897664519586169
Maximum Drawdown = 18.588961245477012

Parameters:
Entry_Threshold = 0.5
Exit_Threshold = 0.5
Lookback Period = 10
Trading Cost = 0.001
Risk Free Rate = 0.05
Metrics:
```

```
# Print the best parameter combination and the corresponding best metric value
print('Best Parameters:')
print('Entry_Threshold =', best_parameters[0])
print('Exit_Threshold =', best_parameters[1])
print('Lookback Period =', best_parameters[2])
print('Trading Cost =', best_parameters[3])
print('Risk Free Rate =', best_parameters[4])
print('')
print('Best Metrics:')
print('Best', target_variable_1, '=', best_metric_1)
print('Best', target_variable_2, '=', best_metric_2)
print('Best', target_variable_3, '=', best_metric_3)
print('Best', target_variable_4, '=', best_metric_4)
```

```
Best Parameters:
Entry_Threshold = 2.5
Exit_Threshold = 0.7
Lookback Period = 14
Trading Cost = 0.001
Risk Free Rate = 0.02

Best Metrics:
Best Portfolio Value = 127777.58168029785
Best Cumulative Return = 27.677581680297852
Best Sharpe Ratio = 2.7042434008195917
Best Maximum Drawdown = 17.349111474591002
```

The result of the sensitivity analysis, analysing the parameter changes (when the other are constants) and their impact on the performance metrics:

- entry threshold, as it increases:
  - ➢ Portfolio Value- can't say definitely, generally INCREASES
  - ➢ Cumulative Return- can't say definitely, generally INCREASES
  - ➢ Sharpe Ratio- can't say definitely, generally INCREASES
  - ➢ Maximum Drawdown- can't say definitely, generally DECREASES
- exit threshold, as it increases:
  - ➢ Portfolio Value- can't say definitely, generally INCREASES
  - ➢ Cumulative Return- can't say definitely, generally INCREASES
  - ➢ Sharpe Ratio- can't say definitely, generally INCREASES
  - ➢ Maximum Drawdown- can't say definitely, generally INCREASES
- lookback period, as it increases:
  - ➢ Portfolio Value- can't say definitely, generally INCREASES
  - ➢ Cumulative Return- can't say definitely, generally INCREASES
  - ➢ Sharpe Ratio- can't say definitely, generally INCREASES
  - ➢ Maximum Drawdown- INCREASES
- trading cost, as it increases:
  - ➢ Portfolio Value- no change
  - ➢ Cumulative Return- DECREASES
  - ➢ Sharpe Ratio- DECREASES
  - ➢ Maximum Drawdown- no change
- risk free rate, as it increases:
  - ➢ Portfolio Value- no change
  - ➢ Cumulative Return- no change
  - ➢ Sharpe Ratio- DECREASES
  - ➢ Maximum Drawdown- no change

The **optimum** combination is,

Optimum entry threshold = 2.5
Optimum exit threshold = 0.7

Optimum lookback period = 14
Optimum trading cost = 0.001
Optimum risk free rate = 0.02

Optimum Portfolio Value = 127777.58168029785
Optimum Cumulative Return = 27.677581680297852 %
Optimum Sharpe Ratio = 2.7042434008195917
Optimum Maximum Drawdown = 17.349111474591002 %

To conclude, research of the pair trading and proper knowledge and implementation of the sensitivity model were vital in completion of this assignment. This report represents the understanding and results of efforts put forward to completing this assignment. The several components of the assignment and how these intertwined with each other and resulted in an output was a major challenge faced during the process. Eventually, the desired end result was obtained with proper implementation of the pair trading strategy, with stocks chosen by analysing the correlation coefficient matrix of NIFTY 50 stocks and formulation of the sensitivity analysis model.